

WEST

Generate Collection

L21: Entry 12 of 41

File: USPT

May 14, 2002

US-PAT-NO: 6389446

DOCUMENT-IDENTIFIER: US 6389446 B1

TITLE: Multi-processor system executing a plurality of threads simultaneously and an execution method therefor

DATE-ISSUED: May 14, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Torii; Sunao	Tokyo			JP

US-CL-CURRENT: 709/100

ABSTRACT:

A program is divided into several instruction streams, and each of them is executed as a thread. A thread processor executed the thread. The thread generates another thread, but one thread is controlled to make a fork operation at most once. Each thread is terminated in the order of generations. A thread manager may be shared with the several thread processors or be distributed to the several thread processors. The thread manager includes a thread sequencer and a thread status table. The thread status table manages execution status of each thread processor and parent-child relation. The thread sequencer requests a thread generation and permits its termination in accordance with the content of the thread status table. The thread processor can execute a thread speculatively.

16 Claims, 24 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 24

WEST

Generate Collection

L20: Entry 4 of 22

File: USPT

Jul 30, 2002

US-PAT-NO: 6427161

DOCUMENT-IDENTIFIER: US 6427161 B1

TITLE: Thread scheduling techniques for multithreaded servers

DATE-ISSUED: July 30, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
LiVecchi; Patrick Michael	Raleigh	NC		

US-CL-CURRENT: 709/102; 709/313, 709/315

ABSTRACT:

A technique, system, and computer program for enhancing performance of a computer running a multithreaded server application. A scheduling heuristic is defined for optimizing the number of available threads. This heuristic alleviates over-scheduling of worker threads by defining a technique to wait to assign an incoming request to a currently-executing thread (upon completion of the thread's current work), instead of awakening a blocked thread for the incoming request. Provision is made to ensure no thread waits too long. Two stages are associated with a passive socket, so that a connection is only bound to a worker thread when work arrives for that connection. A new type of socket is defined, for merging input from more than one source and making that merged input available for scheduling. A giveback function is defined, for optimizing assignment of threads to incoming requests when persistent connections are used. Threads that go idle are put onto an idle queue, releasing them from a worker thread.

30 Claims, 10 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 9

☐

L8: Entry 15 of 59

File: USPT

May 28, 2002

US-PAT-NO: 6397252

DOCUMENT-IDENTIFIER: US 6397252 B1

TITLE: Method and system for load balancing in a distributed object system

DATE-ISSUED: May 28, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Sadiq; Waqar	Rochester Hills	MI		

US-CL-CURRENT: 709/226

ABSTRACT:

One aspect of the invention is a method for load balancing in a distributed object system running on a network comprising a plurality of computers (42, 44) including a first computer (44) wherein the computers (42, 44) are operable to access a plurality of shared objects in a distributed object system. The method comprises instructing an object comprising a part of an application process (50) running on the first computer (44) to record at least one performance statistic in response to a message directed to the object. The application process (50) comprises a multi-threaded process and includes a statistics thread (54). Periodically, at least one performance statistic is obtained using the statistics thread (54) and that performance statistic is sent to a local agent process (48) running on the first computer (44). The performance statistics are relayed to a workload service (46) running on a second computer (42) connected to the network. A new distributed object is instantiated in the memory of one of the plurality of computers (42, 44) based upon performance statistics maintained by the workload service (46).

23 Claims, 2 Drawing figures

Exemplary Claim Number: 1

Number of Drawing Sheets: 1

Set Name Query
side by side

Hit Count Set Name
result set

DB=USPT; PLUR=YES; OP=ADJ

<u>L21</u>	statically near5 threads	41	<u>L21</u>
<u>L20</u>	L19 same l10	22	<u>L20</u>
<u>L19</u>	dynamic\$6 near5 thread\$2	485	<u>L19</u>
<u>L18</u>	simulat\$6 same dynamic\$6 near5 thread\$2	2	<u>L18</u>
<u>L17</u>	simulat\$6 near5 dynamic\$6 near5 thread\$2	0	<u>L17</u>
<u>L16</u>	static\$6 near5 thread\$2 and simulat\$6 near5 dynamic\$6 near5 thread\$2	0	<u>L16</u>
<u>L15</u>	static\$6 near5 thread\$2 and simulat\$6 same dynamic\$6 allocated thread\$2	0	<u>L15</u>
<u>L14</u>	static\$6 near5 thread\$2 same simulat\$6 same dynamic\$6 allocated thread\$2	0	<u>L14</u>
<u>L13</u>	static\$6 near5 thread\$2 and simultat\$6 same dynamic\$6 allocated thread\$2	0	<u>L13</u>
<u>L12</u>	static\$6 near5 thread\$2 same simultat\$6 same dynamic\$6 allocated thread\$2	0	<u>L12</u>
<u>L11</u>	single static\$6 near5 threads	0	<u>L11</u>
<u>L10</u>	static\$6 near5 threads	294	<u>L10</u>
<u>L9</u>	l7 and l8	3	<u>L9</u>
<u>L8</u>	statistic\$6 near5 threads	59	<u>L8</u>
<u>L7</u>	threads near5 simulat\$4	222	<u>L7</u>
<u>L6</u>	threads near5simulat\$4	0	<u>L6</u>
<u>L5</u>	threads same simulat\$4 same state information	5	<u>L5</u>
<u>L4</u>	thread same simulat\$4 and state information	18	<u>L4</u>
<u>L3</u>	thread and simulat\$4 and state information	273	<u>L3</u>
<u>L2</u>	thread same simulat44 same state information	0	<u>L2</u>
<u>L1</u>	6535878	1	<u>L1</u>

END OF SEARCH HISTORY

WEST**Freeform Search****Database:**

US Patents Full-Text Database
US Pre-Grant Publication Full-Text Database
JPO Abstracts Database
EPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Term:**Display:** **Documents in Display Format:** **Starting with Number** **Generate:** ☐ Hit List ☒ Hit Count ☐ Side by Side ☐ Image

Search

Clear

Help

Logout

Interrupt

Main Menu

Show S Numbers

Edit S Numbers

Preferences

Cases

Search History

DATE: Sunday, November 09, 2003 [Printable Copy](#) [Create Case](#)